Requirements for ODP Enterprise Architecture Tools

José Raúl Romero
Dpt. Informática y Análisis Numérico
University of Córdoba
jrromero@uco.es

Antonio Vallecillo
Dpt. Lenguajes y Ciencias de la Computación
University of Málaga
av@lcc.uma.es

Abstract

An important issue to the adoption of any enterprise architectural approach is the availability of tools to support the development, storage, presentation, analysis, improvement and evolution of enterprise architecture representations. As with enterprise architecture methodologies, enterprise architecture tools to support the architectural development process are still emerging. Most important software vendors (e.g., Rational, BEA, IBM, etc.) are investing in the development of large EA tools, mainly focused on their own frameworks or on well-known architectural proposals, such as TOGAF or FEAF. In this paper we identify an initial list of requirements that any tool for EA using RM-ODP should fulfill, as a first step towards the development of the appropriate ODP-Enterprise Architecture tools.

1 Introduction

1.1 RM-ODP

Large-scale heterogeneous distributed systems are inherently much more complex to design, specify, develop and maintain than classical, homogeneous, centralized systems. One way to cope with such complexity is by dividing the design activity according to several areas of concerns, each one focusing on a specific aspect of the system, as described in IEEE Std. 1471 [6]. Following this standard, current architectural practices for designing open distributed systems define several distinct viewpoints. Examples include the viewpoints described in the "4+1" view model [12], the Zachman's framework [25], TOGAF [22], or the Reference Model of Open Distributed Processing (RM-ODP) [7].

In this paper we are interested in the RM-ODP, which is a joint standardization effort by ISO/IEC and ITU-T that creates an architecture within which support of distribution, interworking and portability can be integrated.

Several years after its final adoption as ITU-T Recommendation and ISO/IEC International Standard, the Reference Model of Open Distributed Processing is increasingly relevant, mainly because the size and complexity of current IT systems is challenging most of the current software engineering methods and tools. These methods and tools were not conceived for use with large, open and distributed systems, which are precisely the systems that the RM-ODP addresses. In addition, the use of international standards has become the most effective way to achieve the required interoperability between the different parties and organizations involved in the design and development of complex systems. As a result, we are now witnessing many major companies and organizations investigating RM-ODP as a promising alternative for specifying their IT systems, and for structuring their large-scale distributed software designs. Examples of such projects include the DASIBAO (Démarche d'Architecture des Systèmes d'Information BAsée sur Odp) methodology for specifying IT systems, developed by EDF (Electricité de France) [18]; the Reference Architecture for Space Data Systems (RASDS), developed by the Consultative Committee for Space Data Systems (CCSDS, which includes space agencies such as NASA/JPL, JAXA, ESA, etc.) [4]; the projects developed by the Interoperability Technology Association for Information Processing (INTAP) in Japan (`http://net.intap.or.jp/e/`); or the Synapses project for enabling EU healthcare professionals to share patient records and medical data irrespective of the systems that hold them (`https://www.cs.tcd.ie/synapses/public/`). RM-ODP has also been successfully used for building Financial Systems (see, e.g. [3, 13]) and in Government [20].

The RM-ODP provides five generic and complementary viewpoints on the system and its environment: *enterprise*, *information*, *computational*, *engineering* and *technology*. They allow different participants to observe a system from different perspectives [16]. These viewpoints are sufficiently independent to simplify reasoning about the complete specification of the system. The architecture defined by RM-ODP tries to ensure the mutual consistency among the viewpoints, and the use of a common object model and

a common foundation defining concepts used in all of them (composition, type, subtype, actions, etc.) provide the glue that binds them all together.

1.2 Enterprise Architecture

Software development is a process that can not be seen in an isolated form by companies and organizations, due of the great effort required by current software projects in terms of investments, resources and risks. On the contrary, this process must be considered as a integral element of their current business strategies. The goal of *enterprise architecture* (EA) is to align the business systems and the IT systems in order to improve enterprise competitiveness [24]. Enterprise architecture deals with hierarchical systems that typically span from business entities (e.g., market, department, etc.) down to IT implementation (i.e., technological) entities (e.g. applications, applets, J2EE beans, servlets, DCOM, etc.). In the enterprise architecture, the concept of hierarchical system modeling is crucial. In this sense, RM-ODP is business oriented (e.g., it offers a well-founded enterprise viewpoint) and seems to have the proper elements to model these needs and to obtain the expected benefits from the EA: better return of investment, risk mitigation, flexibility for business grow, more efficient IT operations, etc.

1.3 Tools for Enterprise Architecture

An important issue for the adoption of any enterprise architectural approach is the availability of tools to support the development, storage, presentation, analysis, improvement and evolution of enterprise architecture representations. As with enterprise architecture methodologies, enterprise architecture tools to support the architectural development process are still emerging. And although the RM-ODP is quite a mature proposal, other approaches are gaining ground to RM-ODP in terms of applicability and acceptance. Leading software vendors (e.g., IBM/Rational, BEA, etc.) are investing in the development of large EA tools, mainly focused on their own frameworks or on well-known architectural proposals, such as TOGAF [22] or FEAF [23].

So far, there are no specific commercial tools based on RM-ODP. Several arguments have been put forward by vendors to explain this situation: (a) the small community that comprises the ODP world from a commercial point of view; (b) the lack of bibliographical references and tutorials hinders the use and widespread adoption of RM-ODP in the industry; and (c) the lack of ODP-specific modeling notations and tools. Apparently, (a) can be a consequence of (b) and (c).

Issue (c) has been previously addressed by the ODP community and, in fact, is not new to WODPEC. For example, in [19] the authors studied how RM-ODP naturally serves to capture business needs, system architectures, semantics of processing, choices of technologies, etc. Other proposals (e.g., [15]) discussed the development of EA tools based on the RM-ODP precepts.

In general, these works limit their application to some specific concepts defined by the ODP specification languages, which can be properly used to achieve the proposed objectives in those cases. However, if we wish to build a general tool based on the ODP framework, then we must allow the user to make free use of the whole set of ODP concepts, whose semantics and constraints must guide the tool behavior. In addition, as suggested by Akehurst [1], a model-based tool should allow ensuring the notation- and methodology-independent character of ODP.

In this sense, the increasing interest in the MDA (*Model Driven Architecture*) initiative [17] motivated ISO/IEC and ITU-T to launch a joint project in 2004, which aimed to standardize the use of UML (*Unified Modeling Language*) for ODP system specifications [8]. The goal is that ODP modelers can use the UML notation for expressing their ODP specifications in a standard graphical way, and UML modelers can use the RM-ODP concepts and mechanisms to structure their (large) UML system specifications. ODP languages are expressed in terms of UML Profiles. A direct consequence of this is that a numerous set of commercial (UML) modeling tools become instantaneously available to ODP modelers.

Although these tools provide a concrete notation and make the ODP terminology available to UML modelers, they do not consider other ODP specific aspects. For example, they do not allow enforcing the ODP structuring, conformance and consistency rules. Consistency between viewpoints in not maintained either. And they normally do not deal with multi-viewpoint model management and evolution. Thus, it seems mandatory to build tools whose features are closer to the ODP particularities and, at the same time, that satisfy the actual business needs, so that ODP becomes a competitive alternative for the specification and development of large distributed systems in industrial and business contexts.

In this paper we provide an initial list of requirements that we think any tool for EA using RM-ODP should fulfill. To build such a requirement specification, in Section 2 we first identify the context and stakeholders of any ODP-specific tool. Then, in Section 3, we explore the goals expected in terms of functional and extra-functional requirements. Finally, Section 4 outlines some concluding remarks.

2 Scope and context

Before beginning to collect requirements for any tool, we need to know the target users and the role that they play in the organization. Since we may consider that ODP covers the whole software process, from the business policies and needs to the specific technological platform where the system is implemented, it would be interesting to identify the user roles for the tool and then to capture their expectations.

- [EA] *Enterprise architect*: works with other stakeholders to build a coherent and viable view of the strategies, processes, information and IT resources available in the organization.
- [BA] *Business architect*: aims to improve the business performance by changing and restructuring the work teams, procedures and business tools (e.g., IT systems), if required.
- [PCA] *Process architect*: tries to capture, define, specify, simulate, execute and monitor the business processes.
- [IA] *Information architect*: defines, structures and manages any information comprised by data stored by IT systems, and also discovers and defines both the metainformation and any other existing constraint or interrelationship between data, at the business and system levels.
- [SA] *System architect*: interacts with users and other stakeholders to capture high-level system requirements, based on the needs of the target user and on any temporal or cost constraint. Moreover, this person is in charge of dividing the system into subsystems and components, while ensuring a robust and coherent system architecture.
- [INFA] *Infrastructure architect*: manages the part of the technical architecture that refers to the structured process of designing and building IT (hardware) infrastructure, usually at the enterprise level.
- [INTA] *Integration architect*: defines the mechanisms that allow the cooperation and interoperability between applications, the use of standardized information assets among disparate providers and the discovery of information requirements within the enterprise and IT systems.
- [SP] *Strategic planner*: finds out the way to harmonize IT systems and business policies, infrastructure, resources, research, investments and goals.
- [PGM] *Programme manager*: manages multiple ongoing inter-dependent projects.
- [PM] *Project manager*: manages the resources for a single project in terms of quality assurance, cost and time.
- [RM] *Risk manager*: is in charge of detecting risks and analyzing and developing the proper strategies to manage and mitigate their effects.
- [QAM] *Quality assurance manager*: assures the quality of all business process areas, not just testing.
- [ST] *Software tester*: specifies and executes the process used to help identify the correctness, completeness, security and quality of developed computer software.
- [BAN] *Business analyst*: for a given project, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions.
- [SAN] *Systems analyst*: is responsible for designing, modifying or improving computer information systems by preparing specifications for programmers to follow. This person may also coordinate the development of test problems.
- [PG] *Programmer*: is a software developer, who converts the analysis specifications into executable lines of code or information structures.

RM-ODP offers different useful concepts for each role. These concepts are defined by the different viewpoint languages. Table 1 shows the existing relationship between these roles and each ODP viewpoint, so we are able to identify which views will be more relevant to each user.

3 What should be expected from an ODP tool?

Basically, any ODP tool should fulfill a set of functional requirements that describe the basic functionality that the tool should provide to its users, as well as a set of extra-functional requirements that specify the basic quality characteristics of the tool.

3.1 Functional requirements

Our main premise here is that the tool needs to be *model-based* (i.e., everything should be a model). Under these circumstances, the following list describes the basic set of functional requirements (FR) that we think that any EA/ODP tool should provide.

- [FR-1] The tool should provide model editors for each of the ODP viewpoints. Each editor will implement the corresponding viewpoint language concepts and provide concrete syntax for them.

Table 1. Relationship between roles and ODP viewpoints

Role	EV	IV	CV	NV	TV
Enterprise architect	X	X	X	-	-
Business architect	X	-	-	-	-
Process architect	X	X	X	-	-
Information architect	X	X	-	-	-
System architect	X	-	X	-	-
Infrastructure architect	X	-	-	X	-
Integration architect	X	X	X	X	-
Strategic planner	X	X	-	X	-
Programme manager	X	-	-	-	-
Project manager	X	-	-	-	-
Risk manager	X	-	-	-	-
Quality assurance manager	X	X	X	-	-
Software tester	X	X	X	X	X
Business analyst	X	-	-	-	-
Systems analyst	X	X	X	X	X
Programmer	-	X	-	-	X

- [FR-2] Each viewpoint model editor shall enforce the structuring rules of its corresponding viewpoint language.
- [FR-3] A particular model editor will provide the mechanisms required to model the correspondences between the viewpoint specifications.
- [FR-4] The tool should allow checking the consistency between the different viewpoint specifications w.r.t. the correspondences defined.
- [FR-5] The tool should allow the user to check for *modeling defects* in the individual viewpoint specifications (i.e., models). This is to avoid common defects in the models such as those described in [14]:
  - Methods which are defined, but whose behavior is not described anywhere; e.g., the behavioral specification of an object should include the treatment of all the methods defined in the interfaces it implements. Analogously, all defined methods should be called by some other object (i.e., there are no methods without calls).
  - Objects without names (in, e.g., sequence or activity diagrams).
  - States without names or transitions without triggers in state machines.
  - Methods or events not declared in interfaces.

  This is particularly interesting for the individual viewpoints, for which concrete lists of potential defects need to be identified. (For instance, that all roles types defined in the enterprise viewpoint specification are fulfilled by at least one object type.)
- [FR-6] Models evolve with time. The tool should provide mechanisms to allow change and evolution management. In particular, the tool should allow mechanisms for:
  - Model versioning and version control management.
  - Identification, storage and representation of model differences.
  - Change history. The history needs to be both globally and locally managed. That is, the tool should allow not only managing the global history of the changes made to the ODP system specifications, but also to concentrate on particular contexts only, such as the changes applied to a given viewpoint, or to the correspondences between two given viewpoint specifications.
  - Change enforcement: given a change in a viewpoint set of elements, the tool should provide mechanisms to propagate the required changes in the related elements of the corresponding viewpoints (using the information provided by the correspondences defined between the elements).
- [FR-7] Given the *model-driven* character of the tool, it should provide users with sets of *model transformations* that allow the development of partial implementations of the system. For example,
  - It must be possible to automatically generate, from the information viewpoint specifications,

the appropriate database structure and organization to store the system data. This must be possible for most common implementation platforms and database technologies.

  - The tool should allow the generation of the basic architecture and internal structure of the system implementation for different component platforms (e.g. J2EE, CORBA, etc.), using the information provided by the computational viewpoint specifications.
  - Given the previous partial implementations, the tool may also be able to define the appropriate bridges between the two, e.g., the connections between the component templates and the database APIs.

- [FR-8] The tool should be able to handle repositories of ODP specifications. In this sense, the tool should be able to:

  - store models in repositories (including their different versions);
  - search for ODP system specifications and models that fulfill some selection criteria;
  - retrieve models from the repository (either complete specifications or particular models);

  This would allow system architects to re-use existing ODP specifications and models from previous ODP systems.

- [FR-9] The tool should also take advantage of MDD techniques, allowing:

  - the translation between different notations (if they are based on the same metamodel), and
  - the generation of code from models to different implementation platforms.

- [FR-10] RM-ODP proposes a series of *transparencies*, which imply the use of some well-established implementations of standard solutions to recurrent problems of distributed systems. The tool should allow the automatic addition of such kind of solutions to the ODP specification, to address the transparencies specified by the user.

- [FR-11] Similarly, the tool might provide with a repository of componentized solutions that implement some of the most common *ODP functions* (especially for the partial implementations of the computational and engineering viewpoint specifications).

- [FR-12] RM-ODP is a *coordination framework*. Thus, the tool should provide some extensibility facilities to allow the possibility of adding new features and functionality according to further ODP standards (e.g., IDLs [10], trading services [9], etc.).

- [FR-13] Software metrics are a well established technique for evaluating the quality of software systems assets, from design models to final implementations [5]. Metrics can also be very useful to the software architects to evaluate design alternatives [11], to the project planner to estimate costs and efforts, etc. Thus, the tool should provide the implementation of several sets of metrics for evaluating all these aspects.

- [FR-14] Similarly, the repository of ODP system specifications and models that the tool manages can provide a useful source of information. In general, the repository is expected to contain a fair amount of relevant data for the business (in terms of models, elements, assets, meta-information, etc.). Thus, the tool should provide some mechanisms to infer useful information from the repository elements, using data mining or similar techniques.

- [FR-15] Ideally, different stakeholders might be operating simultaneously on the same models in a concurrent and distributed way, i.e., from different physical locations. Thus, the tool should provide support for collaborative development of system specification. This includes ensuring the *correctness of transactions* and the provision of the proper *communication and coordination mechanisms* between the distributed users working concurrently on the same system specifications.

- [FR-16] Although RM-ODP is methodology independent, in order to be useful the tool should provide support for at least one development methodology (e.g., DASIBAO, RUP, ...), or the possibility to configure user-defined methodologies (probably based on any of the existing ones).

3.2 Extra-functional requirements

Extra-functional requirements (EFR) mainly depend on the specific tool design and, thus, should be independently analyzed for each individual tool. However, there are some non-functional requirements that we think need to be fulfilled by any EA/ODP tool.

- [EFR-1: Interoperability with other tools] The tool should *interoperate* with other ODP-based tools. The use of international standards and conventions should be prescriptive.

  - For example, this issue directly affects the way in which models are serialized, exchanged and stored. Although XMI (*eXtensible Metalanguage Interchange*) might be a possible standard solution, different tools normally use different XMI versions or add proprietary extensions, something that hampers their interoperability.
  - At a higher level of abstraction, it would be desirable that the tool could provide transformations or bridges to other EA proposals, such as TOGAF or FEAF. There are some works that propose possible ways of integration and synergies between different EA proposals (see, e.g., [2, 21]), at least at the conceptual level. Implementing such proposals as transformations between their corresponding models might be a difficult issue, but probably worth pursuing in a near future.

- [EFR-2: Usability] The tool should be *easy to use*, i.e. easy to learn and operate, and attractive to beginners. In this respect, the implementation of helpers and wizards is recommended.

- [EFR-3: Close to domain experts] The tool must be *consistent* with the ODP terminology, and be focused on the ODP conceptual framework.

- [EFR-4: Extensibility] The tool should be *extensible*, so it admits the addition of new features and functionalities (e.g., plugins).

- [EFR-5: Security] Since many different stakeholders can take part of the development process, the tool should ensure that the information is accessed in a secured way. This may be achieved by, e.g., using personal privileges for accessing the data, metainformation, models and functionality; using access control lists, passwords, or any other appropriate mechanisms; etc.

4 Conclusions

Despite the maturity and completeness of RM-ODP, the lack of EA tools to support the development and management of ODP system specifications implies a real impediment to its wider adoption in industrial settings.

The recent appearance of a new standard for establishing a concrete syntax for expressing ODP viewpoint specifications based on UML [8] may facilitate the development of EA tools based on ODP.

This paper has proposed an initial set of requirements for such kind of tools with the goal of identifying the work that needs to be done for developing them. Of course, the list that we have presented here does not try to be exhaustive or complete. It is just a proposal from where to start identifying further requirements, and polishing those described here.

In fact, there are some important issues to address, most of them already discussed (but rarely solved) by the Software Engineering community. For example, do the current standards (e.g., UML4ODP, XMI, etc.) really provide the tool interoperability requested? Is it possible to extend current modeling or EA tools to incorporate these new capabilities? How do we model the action semantics or specify certain constraints on models? Since current tools mainly use their own mechanisms, this may hinder the real extensibility and interoperability of ODP-specific applications.

Furthermore, some of the requirements presented here might be too ambitious, and other may require some refinements to make them more concrete. In this respect, a priorization of the requirements seems to be required, or at least the identification of those which need to be mandatory and those which might be optional. But this falls out of the scope of the paper, and may be more appropriate for discussions at the right forums. In this sense, WODPEC might be the right event for framing this kind of discussions, and provides the venue for further refinements and consolidation of the proposal presented here.

Acknowledgements This work has been supported by Spanish Research Project TIN2005-09405-C02-01.

References

[1] D. Akehurst. Proposal for a model driven approach to creating a tool to support the RM-ODP. In *Proc. of the 1st. Workshop on ODP Enterprise Computing at EDOC*, 2004.

[2] C. Armstrong, J. Cerenzia, E. Harrington, P. Rivett, and F. Waskiewicz. *TOGAF/MDA/IC Synergy Project: Integration Proof-of-concept Results*. Object Management Group, May 2007. OMG doc. omg/07-05-01, `http://doc.omg.org/omg/07-05-01`.

[3] O. Bernet and H. Kilov. From box-and-line drawings to precise specifications: Using RM-ODP and GRM to specify semantics. In H. Kilov and K. Baclawski, editors, *Practical foundations of business system specifications*, pages 99–110. Kluwer Academic Publishers, Norwell (MA), USA, 2003.

[4] CCSDS. *Reference Architecture for Space Data Systems (RASDS)*. Consultative Committee for Space Data Systems, `http://public.ccsds.org/review/default.aspx`, Reston (VA), USA, Jan. 2007. CCSDS Red Book 311.0-R-1, `http://public.ccsds.org/review/default.aspx`.

[5] N. Fenton and S. L. Pfleeger. *Software Metrics: A rigorous approach*. International Thompson Computer Press, 2 edition, 1997.

[6] IEEE. *Recommened Practice for Architectural Description of Software-Intensive Systems*. IEEE Standard 1471, 2000.

[7] ISO/IEC. *RM-ODP. Reference Model for Open Distributed Processing*. Geneva, Switzerland, 1997. International Standard ISO/IEC 10746-1 to 10746-4, ITU-T Recommendations X.901 to X.904.

[8] ISO/IEC. *Information technology – Open distributed processing – Use of UML for ODP system specifications*. International Standards Organization, Geneva, Switzerland, 2006. ISO/IEC FCD 19793, ITU-T Recommendation X.906.

[9] ISO/IEC and ITU-T. *ISO/IEC IS 14771 — Open Distributed Processing — Naming Framework*, 1998.

[10] ISO/IEC and ITU-T. *ITU-T Rec X.920 — ISO/IEC 14750 — Interface Definition Language*, 1999.

[11] J.Muskens, M. Chaudron, and C. Lange. Investigations in applying metrics to multi-view architecture models. In *Proc. of Euromicro 2004*, pages 372–379. IEEE CS Press, 2004.

[12] P. Kruchten. Architectural blueprints — The "4+1" view model of software architecture. *IEEE Software*, 12(6):42–50, Nov. 1995.

[13] T. Kudrass. Describing architectures using RM-ODP. In H. Kilov and K. Baclawski, editors, *Practical foundations of business system specifications*, pages 231–244. Kluwer Academic Publishers, Norwell (MA), USA, 2003.

[14] C. Lange, M. Chaudron, and J. Muskens. In practice: UML software architecture and design description. *IEEE Software*, 22(2):40–46, March-April 2006.

[15] L.-S. Le and A. Wegmann. An RM-ODP based ontology and a CAD tool for modeling hierarchical systems in enterprise architecture. In *Proc. of the Second Workshop on ODP Enterprise Computing at EDOC*. IEEE Digital Library, Sept. 2005.

[16] P. Linington. RM-ODP: The architecture. In K. Milosevic and L. Armstrong, editors, *Open Distributed Processing II*, pages 15–33. Chapman & Hall, Feb. 1995.

[17] OMG. *Model Driven Architecture (MDA) Guide*, 2003. OMG doc.ab/2003-06-01.

[18] A. Picault, P. Bedu, J. L. Delliou, J. Perrin, and B. Traverson. Specifying Information System Architectures with DASIBAO — A Standard Based Method. In *Proc. of ICEIS 2004*, pages 254–264, Porto, Portugal, Apr. 2004. INSTICC Publications.

[19] A. . P. G. Serra, S. A. Vicente, D. Karam, and M. Martucci. Architecting frameworks for specific applications with RM-ODP. In *Proc. of the 1st Workshop on ODP Enterprise Computing at EDOC*, 2004.

[20] L. E. Sweeney, E. V. Kortright, and R. J. Buckley. Developing an RM-ODP-based architecture for the defense integrated military human resources system. In *Proc. of WOODPECKER'2001*, pages 110–123, Setubal, Portugal, July 2001. In conjunction with ICEIS'2001, ICEIS Press.

[21] A. Tang and J. Han. A comparative analysis of architecture frameworks. Technical Report SUTIT-TR2004.01, Swinburne University of Technology, Australia, Aug. 2004.

[22] The Open Group. *The Open Group Architecture Framework (TOGAF) version 8.1 Enterprise Edition*, Sept. 2005. `http://www.opengroup.org`.

[23] US Federal CIO Council. *A Practical Guide to Federal Enterprise Architecture version 1.0*, Feb. 2001. `http://www.whitehouse.gov/omb/egov/a-1-fea.html`.

[24] A. Wegmann. On the systematic enterprise architecture methodology (seam). In *Proc. of the ICEIS 2003*, Angers, France, 2003.

[25] J. A. Zachman. *The Zachman Framework: A Primer for Enterprise Engineering and Manufacturing*. Zachman International, 1997. `http://www.zifa.com`.